Supervised machine learning methods in psychology

Rosenbusch, Hannes; Soldner, Felix; Evans, Anthony M.; Zeelenberg, Marcel

Published in:
Social and Personality Psychology Compass

DOI:
10.1111/spc3.12579

Publication date:
2021

Document Version
Publisher's PDF, also known as Version of record

Received: 17 May 2019 | Revised: 19 November 2020 | Accepted: 27 November 2020

DOI: 10.1111/spc3.12579

RESEARCH ARTICLE

WILEY

Supervised machine learning methods in psychology: A practical introduction with annotated R code

Hannes Rosenbusch[1] | Felix Soldner[2] | Anthony M. Evans[1] | Marcel Zeelenberg[3,4]

[1]Department of Social Psychology, Tilburg University, Tilburg, The Netherlands

[2]Department of Security and Crime Science, University College London, London, UK

[3]Department of Social Psychology, Tilburg University, Tilburg, The Netherlands

[4]Department of Marketing, Tilburg University, Tilburg, The Netherlands

Correspondence
Hannes Rosenbusch, Department of Social Psychology, Tilburg University, 5000 LE Tilburg, The Netherlands.
Email: h.rosenbusch@uvt.nl

Abstract

Machine learning methods for prediction and pattern detection are increasingly prevalent in psychological research. We provide an introductory overview of machine learning, its applications, and describe how to implement models for research. We review fundamental concepts of machine learning, such as prediction accuracy and out-of-sample evaluation, and summarize standard prediction algorithms including linear regressions, ridge regressions, decision trees, and random forests (plus additional algorithms in the supplementary materials). We demonstrate each method with examples and annotated R code, and discuss best practices for determining sample sizes; comparing model performances; tuning prediction models; preregistering prediction models; and reporting results. Finally, we discuss the value of machine learning methods in maintaining psychology's status as a predictive science.

1 | INTRODUCTION

Psychologists are increasingly interested in adopting powerful computational techniques from the field of machine learning to accurately predict real-world phenomena (see Yarkoni & Westfall, 2017). The current work introduces machine learning as a collection of methods and tools that can be used in prediction. We review fundamental concepts of machine learning, discuss its relationship with standard psychological methods, and give

concrete guidelines to implement machine learning projects in R (Core Team, 2018) using the *caret* package (Kuhn, 2015). The annotated R-code is presented in several boxes throughout the paper. Our example analyses in the tutorial sections can be replicated with data and scripts provided in the Supporting Information. Finally, we provide recommendations for best practices and warn of common dangers when implementing machine learning techniques.

2 | MACHINE LEARNING AND PREDICTION ACCURACY

Machine learning is "a set of methods that can automatically detect patterns in data, and then use the uncovered patterns to predict future data" (Murphy, 2012, p. 1). Psychologists already have the tools to detect patterns in data; most commonly ordinary least-squares (OLS) regression techniques. When developing and testing theories, such patterns (often regression coefficients) are examined for statistical significance to ascertain the effect of predictors on outcome variables. The remaining element of machine learning, predicting future data, is becoming increasingly important to social scientists (Alharthi, Guthier, & El Saddik, 2018; Kübler et al., 2020; Plonsky, Erev, Hazan, & Tennenholtz, 2017; Walsh, Ribeiro, & Franklin, 2017).

When the primary goal is an accurate prediction, researchers can again use statistical models to link predictor variables to outcome variables. The search for accurate models lies at the heart of machine learning. Now, the primary metrics of interest are no longer model coefficients, but the accuracy of the model's predictions (i.e., how close predictions are to observed values). Importantly, model accuracy must be evaluated using new data. Machine learning models are fit on one data set (the "training set"), and predictions are made and evaluated using a new data set (the "test set"). This out-of-sample testing avoids overestimating a model's accuracy, as models are generally overfitted to training samples.

Testing out-of-sample prediction accuracy is sufficient to turn common regression analyses into machine learning. Previous work found, for instance, that insufficient sleep is associated with suicidal intentions (Ribeiro et al., 2012). This work, based on inferential tests, reveals a significant correlation between sleep deprivation and suicidal thoughts. However, it does not tell us how *accurately* sleep predicts suicidal intentions. Is the model accurate enough to implement alert systems based on sleep quality? A more recent publication featured a machine learning study, focused on prediction accuracy, in which the risk of suicide attempts was predicted using an array of psychological variables (Walsh et al., 2017). However, the project's prediction models did not include sleep quality as a predictor. The potential contribution of sleep quality in a model for accurately predicting suicide attempts, therefore, remains unquantified.

Psychology and machine learning come together whenever the research question is "How well does x predict y?" Most psychologists use linear regression to quantify achievable prediction accuracy (e.g., Hassan, Shiu, & Shaw, 2016; Rimfeld, Kovas, Dale, & Plomin, 2016). However, machine learning offers a wide range of alternative models that usually lead to substantial accuracy improvements (e.g., Joel, Eastwick, & Finkel, 2017; Park et al., 2015; Plonsky et al., 2017; Wang & Kosinski, 2018). The cited projects can be labeled "applied machine learning" and differ from most psychological work in three ways:

1. A focus on prediction accuracy.
2. Measures of prediction accuracy for *new* samples.
3. Use of prediction models that, unlike typical linear regressions, are manually tuned to better fit the specific problem at hand (see the Section 3.1.1.).

These three aspects will be described in more detail throughout the text, and always about an example research question.

2.1 | Example: Predicting regional differences in happiness

For demonstration purposes, we focus on an example research question: "How well can we predict county-level happiness (i.e., regional averages) in the United States?" We start with prediction models familiar to most psychologists, linear and logistic regressions, in which outcome scores (i.e., county-level happiness) are expressed as mathematical combinations of predictor scores. The standard regression approach is to estimate an equation that minimizes the distances (residuals) between the original data points and the values predicted by the regression line. We use these familiar models to introduce the concepts of prediction accuracy and out-of-sample evaluation. Subsequently, we introduce three of the most common machine learning models (plus three in the Supporting Information) and provide sample R code to implement these models. For all code sections, we utilize the R package *caret* (Kuhn, 2015), which includes a large and standardized selection of prediction models. A package following tidyverse principles for machine learning is *tidymodels* (Kuhn & Wickham, 2020).

2.2 | Prediction accuracy

The central premise of supervised machine learning is to use statistical models to make (accurate) predictions.[1] In the following section, we describe the most relevant metrics for assessing accuracy. Our outcome of interest is county-level happiness (BRFSS, 2005-2010; see Supporting Information) and our predictor variable is the relative number of people drinking alcohol (measured at the county level; e.g., Bellos et al., 2013). We split the data into two data frames, using the first for fitting ("training") the regression model and the second for testing its accuracy (see Section 2.3 below).

```
> #The dataframe "trainset" contains the data for 1911 counties
> # The dataframe "testset" contains the remaining 635 counties
> dim(trainingset)
[1] 1911    2
> dim(testset)
[1] 635   2

> #We work with two variables: alcohol use, and happiness
> colnames(trainingset)
[1] "alcohol_use" "happiness"

#We fit the prediction model with caret's train function.
#This function can be used to fit a large selection of machine learning models.
#Here we chose the model "lm", which specifies a linear regression model.
predictionmodel = train(happiness ~ alcohol_use, data = trainingset, method = 'lm')
```

2.2.1 | Prediction accuracy for continuous outcomes

When the predicted outcome is continuous there are multiple approaches to assess accuracy. One of the most common metrics is R^2, the proportion of variance accounted for by the model. Higher R^2 values signify higher accuracy. When the residual variance (i.e., "sum of squared residuals" in the formula below) is zero, the model makes perfect predictions and $R^2 = 1$. If the summed residuals are equal to the total variance (in the denominator), the model is useless, predicting the mean is equally accurate, and $R^2 = 0$.

$$R^2 = 1 - \frac{\text{Sum of squared residuals}}{\text{Sum of squared deviations from the mean}}$$

```
> #We evaluate the prediction accuracy on the test set with caret's R2 function.
> predictions = predict(predictionmodel, testset)
> R2(predictions, testset$happiness)
[1] 0.092
```

The linear regression model based on county-level alcohol consumption predicted 9.2% of the variance in regional happiness.[2] Equivalently, researchers could measure model accuracy by correlating predicted and observed scores (Youyou, Kosinski, & Stillwell, 2015).

```
> cor(predictions, testset$happiness)
[1] 0.303
```

Other metrics focus on the average residual size (instead of residual proportion as in R^2) with smaller residuals signifying higher prediction accuracy. Most common are the mean absolute error (MAE; negative signs of residuals are removed) and the root mean square error (RMSE; residuals are squared).

$$\text{MAE} = \frac{1}{n} \times \sum_{i=1}^{n} |y_i - \hat{y}_i|$$

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)}{n}}$$

```
> MAE(predictions, testset$happiness)
[1] 0.0518286
> RMSE(predictions, testset$happiness)
[1] 0.07506596
```

The MAE-metric weighs each prediction error equally, whereas the RMSE gives more weight to large errors (due to the squaring). This makes the MAE metric easy to interpret, while the RMSE is more relevant when large mispredictions are disproportionately costly. Evaluating either metric requires familiarity with the scale of the outcome variable.

2.2.2 | Prediction accuracy for categorical outcomes

When the outcome variable is categorical (e.g., predicting whether a county is in the top or bottom 50% in terms of happiness) there are again multiple options for assessing accuracy. Many measures can be extracted from the so-called confusion matrix, which shows the cross-tabulation of predicted and observed values (see Table 1 for an example).

The most prominent measure for categorical data is the accuracy score (or just "accuracy") and is defined as the number of correct predictions divided by the total number of predictions (i.e., the proportion of correct predictions):

$$\text{Accuracyscore} = \frac{\text{Truepositives} + \text{Truenegatives}}{\text{Truepositives} + \text{Truenegatives} + \text{Falsepositives} + \text{Falsenegatives}}$$

More refined quantifications of accuracy pertain to the number of false and true positives and negatives and are known as sensitivity (also called "recall") and specificity.

TABLE 1 Example of a confusion matrix

	Predicted value: Positive	Predicted value: Negative
True value: Positive	True positives	False negatives
True value: Negative	False positives	True negatives

Note: True positives were correctly predicted to be happy above-average, false positives were incorrectly predicted to be happy above-average. True (vs. false) negatives were correctly (vs. incorrectly) predicted to score below-average.

```
> caret::confusionMatrix(categ_predictions, testset$categ_happiness)
Prediction FALSE TRUE
    FALSE   203  116
    TRUE    123  193
        Accuracy : 0.6236
        Sensitivity : 0.6227
        Specificity : 0.6246
```

Sensitivity signifies how many of the positive measurements (here: above-average counties) were predicted to be positive cases, whereas specificity describes how many of the negative measurements (here: below-average counties) were predicted to be negative cases (cf. signal detection theory; Macmillan, 2002). The "precision" refers to the number of true positives divided by all positive predictions.

$$\text{Sensitivity} = \text{Recall} = \frac{\text{True positives}}{\text{True positives} + \text{False negatives}}$$

$$\text{Specificity} = \frac{\text{True negatives}}{\text{True negatives} + \text{False positives}}$$

$$\text{Precision} = \frac{\text{True positives}}{\text{True positives} + \text{False positives}}$$

Thus, if researchers want to prioritize that their model is accurately predicting "positive" cases (here: labeling happy counties as such), they might want to focus on the model's sensitivity. If false positives are to be avoided, then the model's precision becomes more important. When priorities are not one-sided, the harmonic mean between precision and recall can be used (i.e., the F1 score; Scherer, Stratou, Gratch, & Morency, 2013).

$$F1\ \text{score} = 2 \times \frac{\text{Precision} \times \text{recall}}{\text{Precision} + \text{recall}} = \frac{2 \times \text{Truepositives}}{2 \times \text{Truepositives} + \text{Falsepositives} + \text{Falsenegatives}}$$

```
> caret::confusionMatrix(categ_predictions, testset$categ_happiness, mode = "prec_recall")
F1 : 0.6295
```

Given the many alternatives, it is always preferable to report multiple accuracy metrics (e.g., by plotting false positive against true negative rates). Focusing on one individual metric can distort the reader's impression of accuracy and hinders comparisons between projects (e.g., Akosa, 2017).

2.3 | Out-of-sample evaluation

In machine learning, accuracies are not tested on the same sample that was used to fit the prediction model (see the distinction between training and testing above). Accuracy metrics based on training data would be biased due to

"overfitting," meaning that parameters in the prediction model not only reflect the true relationships between predictors and outcome, but also idiosyncratic sample characteristics. In other words, prediction models are optimized to predict an outcome from predictor variables *in the present sample*, which is straightforward with small samples and multiple predictor variables. One could, for instance, predict the voting behavior of ten people perfectly, using their clothing selections and favorite breakfast cereals ("if a person wears a red, worn-out Nike sweater, and likes Fruit Loops, predict Democrat"), but applying the same model to ten new people will likely result in no accuracy at all because patterns exploited in the model-fitting data do not generalize to the model-testing data.

Accuracy scores should quantify the accuracy that can be expected when applying the model to new data. Thus, the ideal way to test accuracy is to fit the model on one sample (training set) and evaluate the model on a different sample (test set). We discuss two of the most common approaches to obtain training and test sets in machine learning.

2.3.1 | Train–test split (hold-out method)

The simplest way to generate separate training and testing samples is to collect one big sample and then randomly split it (which is how we obtained the data frames above).

```
#We randomly selected 75% of the row numbers of the full data.
indices = createDataPartition(data$happiness, p = 0.75, list = FALSE)

#We included the selected rows in our training set (i.e. the data that we fit the model on).
trainingset = data[indices,]

#The remaining rows go into the test set (i.e. the data that we use to evaluate the model accuracy).
testset = data[-indices,]
```

An alternative (superior) method is to collect one sample for training, fit the model on this sample, preregister the model, and then collect a new sample for the test set (cf., Brandt, 2017). We provide an example for preregistering machine learning models in the Supporting Information, which can be used as a template.

When splitting the original sample, how much data should be used for model fitting, and how much should be used to quantify the accuracy? Commonly, the training set encompasses 60%–80% of the data, while the test set has 40%–20% (e.g., Ng, 2018). Considering the purpose of the test set is helpful when making cut-off decisions. That is, how many predictions does the researcher want to see before judging the model? A small test set can already inform the researcher whether the model is close to perfect or close to useless. If more precise evaluations are needed, then the test set needs to be larger. Another helpful heuristic is that the test set should be representative of the target population, which limits how small it can be. For more detailed considerations see the Section 2.5 below.

2.3.2 | K-fold cross-validation

Randomly splitting data into a training and test set bears risks, especially if the original sample is relatively small. Random chance could affect the transferability of the prediction model (and the computed test accuracy). A process called k-fold cross-validation offers a partial solution: this process involves *repeatedly* splitting the data, fitting the model, and testing it on new data. In k-fold cross-validation, the data is split into k (often $k = 10$; Kuhn & Johnson, 2013) equally-sized subsets called folds. Subsequently, k rounds of model fitting, test set prediction, and accuracy evaluation are conducted (see Figure 1). In the first round, the model is fit on the first 9 subsets, and the model is used to make predictions for the 10th subset. In the second round, the model is fit on subset 1 through 8 plus subset 10 and evaluated on subset 9, etc. The 10 individual accuracy quantifications obtained from this procedure are averaged to give an estimation of the model's overall accuracy. This procedure mitigates the danger of chance affecting test accuracy; permits researchers to use relatively more of their data for training the model; and highlights variability in the model's accuracy.

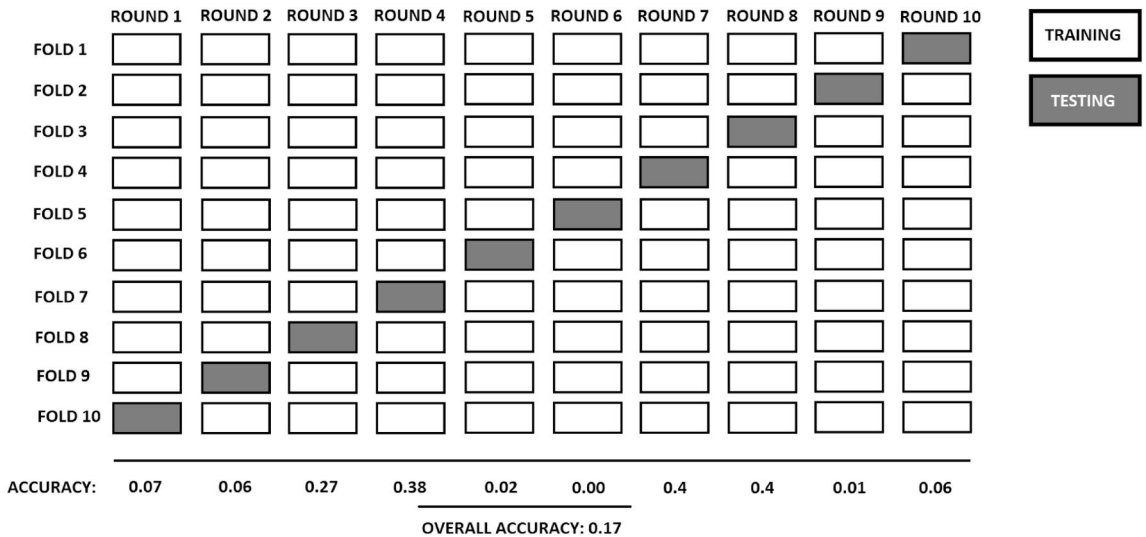


FIGURE 1 The process of cross-validation, including random splitting of the dataset into k (here 10) folds, fitting of the model on $k - 1$ folds, and computing the accuracy achieved on the held-out fold. After k rounds of training and testing, the average accuracy can be computed alongside the variability of the achieved accuracy

For our example case of predicting happiness with alcohol use, we can easily implement this procedure instead of the simple train–test split.

```
> #We prespecify that the following cross-validation should have k=10 splits.
> cross_validation = trainControl(method="cv", number=10)

> #We again train the model.
> #We do not need to explicitly declare training and test set.
> #caret automatically implements the cross-validation.
> predictionmodel = train(happiness ~ alcohol_use, data = data, method = 'lm', trControl =
cross_validation)

> #We obtain a list of results obtained for each split.
> predictionmodel$resample
   RMSE      Rsquared   MAE       Resample
1  0.073     0.077      0.055     Fold01
2  0.086     0.040      0.055     Fold02
3  0.073     0.083      0.050     Fold03
4  0.084     0.089      0.055     Fold04
5  0.078     0.028      0.052     Fold05
6  0.070     0.167      0.051     Fold06
7  0.069     0.129      0.051     Fold07
8  0.062     0.182      0.049     Fold08
9  0.074     0.170      0.051     Fold09
10 0.070     0.152      0.053     Fold10

> #We obtain the overall (average) result.
> predictionmodel$results
   RMSE      Rsquared   MAE       RMSESD    RsquaredSD  MAESD
   0.074     0.112      0.052     0.007     0.056       0.002
```

In this case, the result of the cross-validation procedure is in line with our initial training and test split. However, there is some variability in accuracy estimates. In one of the 10 subsets, the model only explains 4% of the variance (Fold 02) whereas in another it explains 18% (Fold 08). Being able to observe such variability is one reason why it is preferable to conduct k-fold cross-validation instead of a train–test split, as the latter should only be applied with very large sample sizes. The disadvantage of k-fold cross-validation lies in an increased need for computational resources, which should be transparently described when discarding k-fold cross-validation.

2.4 | Interpreting prediction accuracy

Interpreting prediction accuracy metrics (e.g., as high or low accuracy) depends on the context and requires domain expertise. There are different ways of evaluating prediction accuracy, and most involve comparisons with a reference point. For psychologists, the reference point is often the measurement reliability of the outcome variable, which indicates the maximum possible accuracy. When the outcome's reliability is $r = .7$, prediction cannot exceed this ceiling without predicting random error variance. This upper limit entails, for instance, that demographic info (often measured with perfect reliability) can usually be predicted more accurately than noisy personality scores (Kosinski, Bachrach, Kohli, Stillwell, & Graepel, 2014). Thus, accuracy scores close to the outcome's reliability are considered high.

Baseline (or null) models are another common standard for comparison: Simple baseline accuracies for continuous variables might be provided by the model $\hat{y} = \text{mean}(y)$, and for categorical variables by $\hat{y} = \text{mode}(y)$. If a new model cannot predict y substantially better than such baseline models, the performance of the new model can

be evaluated as poor. Imagine having a model predicting relapse rates among drug addicts with an accuracy of 80.3%. This accuracy is not impressive if only 20% of investigated people relapse. Always predicting the mode ("no relapse") would already lead to an accuracy of 80% and the new model barely improves on that. Still, if there are no better alternatives (e.g., human judgment), an increase of 0.3% accuracy can amount to saving many lives. That is to say, context matters for evaluating accuracies (Sumner, Byers, Boochever, & Park, 2012).

Lastly, it is common to consider prior prediction attempts. Such reference points are commonly discussed in the machine learning literature, where the history of classic prediction challenges receives much interest (e.g., identifying words from speech or tumors on scans). If a model can improve historical accuracies, potentially including the accuracy of human raters (Youyou et al., 2015), it is argued to be relatively accurate. While psychological research has yet to develop a set of classic prediction problems, many challenges would benefit from an ongoing competition and bookkeeping. Examples include the behavior of people in economic games (e.g., Plonsky et al., 2017), health trajectories (e.g., Chekroud et al., 2016), or dispositional traits (e.g., Bachrach, Graepel, Kohli, Kosinski, & Stillwell, 2014).

2.5 | Sample sizes in machine learning

In machine learning, complicated models with many coefficients might require millions of observations (He, Zhang, Ren, & Sun, 2016), while other models can be fit using fewer than a thousand cases (Golbeck, Robles, & Turner, 2011). An accessible description of factors affecting required sample sizes in machine learning is provided by Raudys and Jain (1991). However, there are no exact guidelines. Here we present four strategies to help determine appropriate sample sizes:

1. Becoming familiar with similar research projects and making an overview containing each project's (a) predictor variables, (b) outcome variable, (c) prediction model(s), (d) sample size, and (e) achieved prediction accuracy. Projects with similar predictors, outcomes, and models usually indicate how accurate models will be given specific sample sizes. We provide a short example in Table 2.
2. Look for similar data published online. In machine learning, it is common to publish data sets to stimulate prediction competition and knowledge exchange. Acquiring a data set allows you to plot achieved accuracies of specific models against the sample size used for training the model (i.e., learning curves). The saturation point of such curves can give you a good reference for how much data you might need (Figueroa, Zeng-Treitler, Kandula, & Ngo, 2012).
3. Simulate a realistic data set based on prior knowledge on variable distributions. Multiple packages in R are available to generate data sets with prespecified covariance structures (e.g., Goldfeld, 2018). Examining how the accuracy of prediction models changes with different sample sizes (see Figure 2), can help in estimating realistic accuracy levels. Note that this approach requires a priori assumptions about the data and is less feasible when there are many predictors.
4. Collect initial data to assess model requirements. Researchers can diagnose a lack of training data by investigating how the achieved testing accuracy compares to the training accuracy. If a model's training accuracy is much higher than its testing accuracy, the model is too complex for the size of the training set, and therefore it is severely overfitted to the training data (i.e., a high variance problem). In such instances, it is reasonable to collect more data (or reduce model complexity; e.g. changing from nonlinear to linear models). If both training and testing accuracy are poor, researchers can diagnose that their model is underfitted (i.e., a high bias problem) and needs a higher degree of complexity or better predictors.

In summary, machine learning projects split data into separate training and testing sets. As the shape of the best prediction model is determined based on the data, it is more difficult and less common to preregister required sample sizes. However, machine learning requires extra data for out-of-sample evaluation and prediction models

TABLE 2 Example papers for predicting extraversion from Facebook material

Paper	Predictor	Outcome	Model	Accuracy	Sample size
Kosinski et al., 2014	Facebook friends, groups, likes, network density, photo tags, statuses	Extraversion	Linear regression	$r = .31$	16,900
Schwartz et al., 2013	Facebook language	Extraversion	Ridge regression	$r = .38$	75,000
Park et al., 2015	Facebook language	Extraversion	Ridge regression	$r = .42$	66,732

often include many predictors/coefficients; as a result, required sample sizes are typically large. The following section introduces popular machine learning models, including how they can be implemented in R code, and "tuned" to improve accuracy.

3 | COMMON MODELS IN MACHINE LEARNING

Traditional regression methods, as used above, are often not expected to constitute machine learning. However, they are the backbone of many machine learning techniques and should be used as a reference for prediction accuracy (Jie, Collins, Steyerberg, Verbakel, & van Calster, 2019). When it comes to boosting the achievable accuracy, it is advisable to consider other methods, some of which we introduce here and in the Supporting Information.

3.1 | Ridge regression

In standard linear regression models, the individual beta coefficients are optimized to reduce the residual sum of squares of the outcome variable. This optimization should improve accuracy, but also guarantees that the coefficients are perfectly tailored to the present sample, which might diminish generalizability to other samples. Coefficients for each predictor variable are precisely specified (to the last decimal) and even predictors that are not truly related to the outcome variable virtually always have a non-zero coefficient because they *happen* to spuriously explain a small part of residual variance (Lever, Krzywinski, & Altman, 2016). The result is an overfitted model which will not perform well with new samples. This problem becomes exacerbated if the number of predictor variables is high in relation to the number of observations, and if predictor variables are correlated (Askin, 1982). In such cases, the high-dimensional prediction model can explain a large proportion of variance in the training sample, but is likely too complex for the sample size, and achieves poor accuracies on new samples (Dana & Dawes, 2004). The need for much more data for small increases in model complexity is referred to as the "curse of dimensionality."

A potential solution is to use an alternative procedure that simultaneously minimizes both the residual variance *and* model complexity. Ridge regression models accomplish these goals by biasing individual regression coefficients toward zero. Hence, ridge regression models give more weight to the intercept, making predictions less variable and decreasing the magnitude of prediction errors that emerge from poor model transferability. This simplified model has less noise and can be more generalizable.

Statistically, ridge regression suppresses beta coefficients by changing the optimization criterion of the standard regression model. In addition to minimizing residual variance, the model also attempts to minimize the sum of squared beta coefficients. The result is a tradeoff between residual reduction and complexity reduction, which is expressed in the cost function to be minimized:

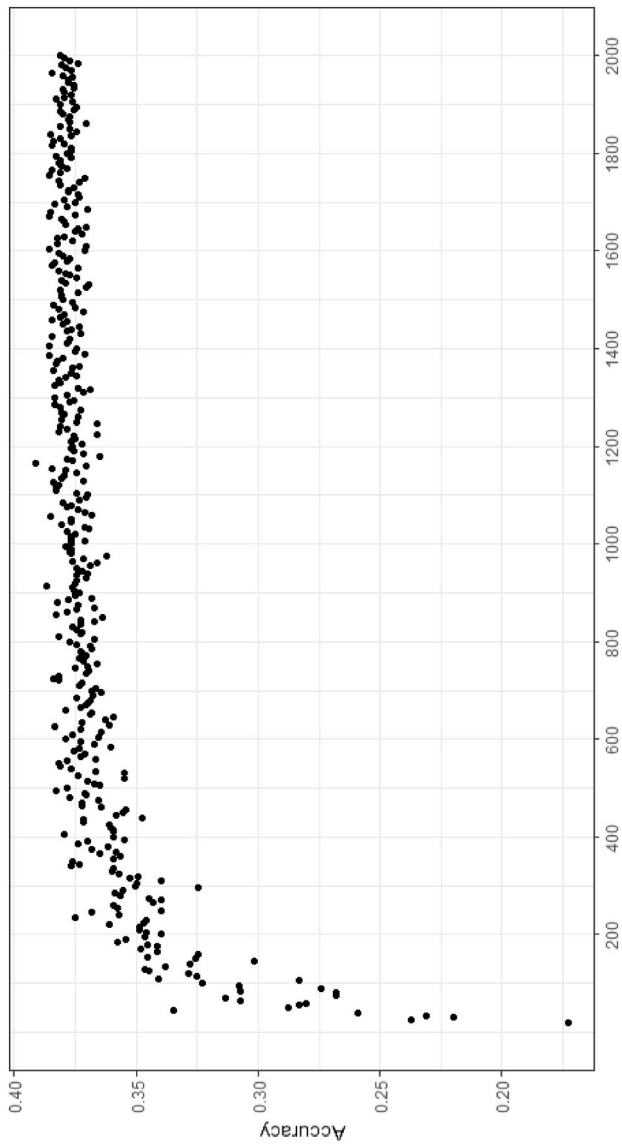


FIGURE 2 The "learning curve" of a linear regression model, based on simulated data with 6 predictor variables (inter-correlations between .15 and .70; see Supporting Information for code). Accuracy is computed as the correlation between predicted and observed values. The learning curve suggests that around 600 observations are needed for approaching the maximal prediction accuracy with this model

$$\text{Cost linear regression} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

$$\text{Cost ridge regression} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda * \sum b^2$$

The ridge regression cost function has a parameter λ (lambda) that is absent in standard linear regression. Lambda is an example of a *hyper*parameter (see next section), that needs to be manually tuned using data. Lambda quantifies how strongly regression weights should be suppressed, thereby dictating the tradeoff between error reduction and model simplicity. Higher values of lambda lead to a stronger penalization of regression weights and therefore a more restricted/conservative model. On the other hand, when lambda is zero, ridge regression becomes identical to the standard OLS regression. It is standard practice to test a range of values for lambda (either manually or automatically) and choose the value that gives the best accuracy for new cases, but not yet the final test set (Claesen & De Moor, 2015). Next, we describe the general practice of hyperparameter tuning and demonstrate how the specific hyperparameter lambda can be tuned for ridge regression models.

3.1.1 | Hyperparameter tuning

For linear regressions, researchers can fit models without having to manually specify further parameters. That is to say, if the data are the same, two researchers fitting a regression model should usually obtain the same result. Machine learning techniques (like ridge regression) are often tuned with hyperparameters, which are not automatically tuned by the data. Hyperparameters determine stable model characteristics *before* the model is fit to a dataset. There are some parallels in traditional psychological methods. For example, in factor analyses, researchers can specify model structure a priori (e.g., four factors should be extracted).

Hyperparameter settings are used to increase prediction accuracy. Often, they determine model complexity (vs. parsimony), or the procedure with which a model incorporates new data and adjusts its coefficients. Most prediction models use a small set of hyperparameters. Tuning them involves testing a range of possible values and selecting the value with which the model achieves the highest out-of-sample accuracy. If multiple hyperparameters are tuned simultaneously, researchers typically test ranges for each hyperparameter, and try out different combinations (i.e., the "value range search" becomes a "value grid search"). There are no set rules for which and how many values should be tried out. Generally, it makes sense to try out a wide range of values, for instance, by specifying value ranges and letting software pick random numbers within that range (Bergstra & Bengio, 2012).

Importantly, the model with the best hyperparameter values is ultimately evaluated again on new data. Most commonly, all models are fit on sample A, and the final model is selected based on its accuracy achieved on sample B. The accuracies achieved on sample B might still overestimate a model's true accuracy because we select the best model out of a potentially wide range of models. Said differently, we might "manually" overfit to sample B during hyperparameter selection. Thus, after fitting the models on sample A (the training set), and selecting the best performing model on sample B (the development set), researchers apply the final model to a new sample C (the test set). Notice, that we only have to introduce the development set (sample B), if we tune model hyperparameters. In the case of linear regression above, for instance, we only had one model, allowing us to directly quantify prediction accuracy on the test set (see Figure 3).

3.1.2 | Implementation of ridge regression

Ridge regression is useful if researchers have a large number of (intercorrelated) predictors relative to their sample size. An example case is to predict psychological phenomena based on language (i.e., where the frequencies of

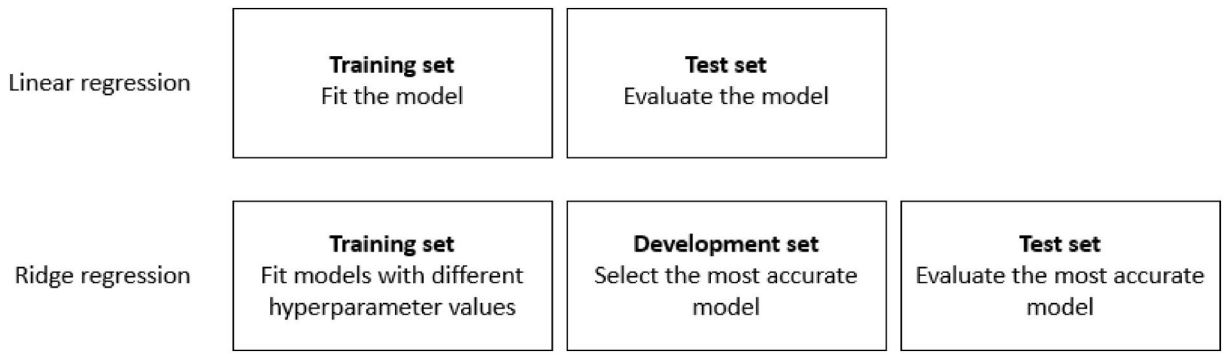


FIGURE 3 The difference in training and test split between OLS regression and models with tunable hyperparameters. After determining the best model settings through the development set, the chosen model is often refitted on all nontest data (training and development) before being evaluated on the test set(s)

specific words from texts are used as predictors). A regression model that predicts an outcome based on language often has as many predictor variables as there were unique words in the dataset. Further, many words are highly correlated in their usage leading to collinearity and unstable model coefficients. These conditions make ridge regression a sensible choice (Schwartz et al., 2013; Zhang & Oles, 2001).

We demonstrate how to implement ridge regression by using regional Twitter language to predict regional happiness (Wang et al., 2014). Each predictor variable pertains to the relative frequency with which a specific word is used by Twitter users in the target region, meaning that the model includes a total of 25,902 predictors. In other words, we ask if regional differences in word use on Twitter can be used to predict regional differences in happiness.

Implementing the ridge regression model is similar to linear regression. However, there are three important differences. First, we implement cross-validation to split the data into three sets, training, development, and test (in linear regression, we only need training and test sets). Second, we tune the hyperparameter lambda during cross-validation using the training data.[3] Third, when training the model, we additionally specify that predictors should be standardized (i.e., centered and scaled) before the model is fit. Given that we penalize high beta coefficients, all predictors should be on the same scale.

```
> #Split off the test set from the training and development data.
> indices <- caret::createDataPartition(data$happiness, p = 0.75, list=FALSE)
> train_dev_data <- data[indices,]
> testdata <- data[-indices,]

> #We implement 10 fold cross-validation (same as above).
> cross_validation <- trainControl(method="cv", number=10)

> #Set possible values for hyperparameter lambda.
> tuning <- expand.grid(alpha = 0, lambda = c(0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4))

> #We train the model (including preprocessing).
> #The cross-validation is repeated for each value that the hyperparameter can take on.
> predictionmodel <- train(happiness ~ ., data = train_dev_data, method = 'glmnet',
preProcess = c("center", "scale"), tuneGrid = tuning , trControl = cross_validation)
        Fitting alpha = 0, lambda = 1.6 on full training set

> #Get the results.
> predictionmodel$results
   lambda  RMSE   R2     MAE
1  0.1     0.067  0.104  0.049
2  0.2     0.066  0.106  0.048
3  0.4     0.065  0.111  0.048
4  0.8     0.065  0.114  0.047
5  1.6     0.065  0.115  0.047
6  3.2     0.065  0.111  0.047
7  6.4     0.065  0.101  0.048

> #Make predictions with best model and evaluate accuracy
> test_predictions <- predict(predictionmodel, testdata)
> R2(test_predictions, testdata$happiness)
[1] 0.122
```

The most accurate model has a lambda value of 1.6. Notice that caret automatically selects this value and uses it to fit the final model on all the nontest data (the training and development sets). Subsequently, we get an R^2 value for the test data which is slightly higher than in the training/tuning phase. The reason is likely that the final model was fitted on all training and development observations, whereas before, we used a cross-validation approach and therefore 10% less data for model fitting. A traditional OLS regression without penalization achieves an accuracy of $R^2 = .11$ using the same predictors, which is just marginally worse. However, some implementations of linear regression in R throw warnings and give worse results when the number of predictors is much higher than the number of observations. Example implementations of ridge regression in psychological research are provided by a range of authors (Dana & Dawes, 2004; Eichstaedt et al., 2015; Ghandeharioun et al., 2017).

3.1.3 | Alternatives to ridge regression

Some techniques offer similar approaches to ridge regression, the most prominent example being LASSO regression. The sole difference between both model types is that LASSO weight penalization is applied to weight magnitudes rather than squared weights. The close relation between both approaches naturally leads to the question of which model to choose for a given problem. While there are possible theoretical considerations (such as using LASSO for predictor selection, Tibshirani, 1996), many researchers treat this decision as a tuning problem by choosing the model that appears to be more accurate. It is even possible to employ hybrids of both approaches (elastic net regression) and finetune the contribution of each approach to weight penalization.

3.2 | Decision trees and random forests

Decision trees are another predictive algorithm and (similar to OLS regression) it serves as the backbone for very powerful prediction models like random forests (introduced next).

Decision trees split observations into increasingly homogeneous subgroups based on binary questions about predictor values (e.g., "Does this observation score lower than 85 on the questionnaire?"). Accordingly, the decision tree consists of a sequence of questions used to subcategorize the data set. Individual observations are funneled through the tree, and at each crossway of branches, the observation is either guided to the left or right depending on whether the answer to the specific binary question is yes or no. The tree's endnotes are called leaves and contain homogenous subsets of the training data. Predictions are made by funneling a new case through the tree and assigning the value that was most common in the training data that landed on the same leaf ("majority vote"). For continuous variables, the predicted value is usually the mean of all training observations on this leaf.

The sequence of questions, as well as the optimal cut-off values within each branching fork, are determined when fitting the tree model to the training data. Intuitively, it is favorable to have a tree model that leads the same outcome values to land on the same leaves and dissimilar values on different leaves. Said differently, if a leaf only includes observations that have the same outcome value (homogenous), we can expect new cases landing on this leaf to also have that (or a very similar) value. Conversely, if a leaf includes observations that have a range of very different values (heterogeneous), we can be less certain about predictions for new cases landing on this leaf. Thus, the statistical criterion which is minimized when fitting a tree model is the heterogeneity of measurements in each of the final leaf nodes. This measure of heterogeneity is often an entropy score. For the hypothetical example tree in Figure 4 the formula the entropy in leaf i is:[4]

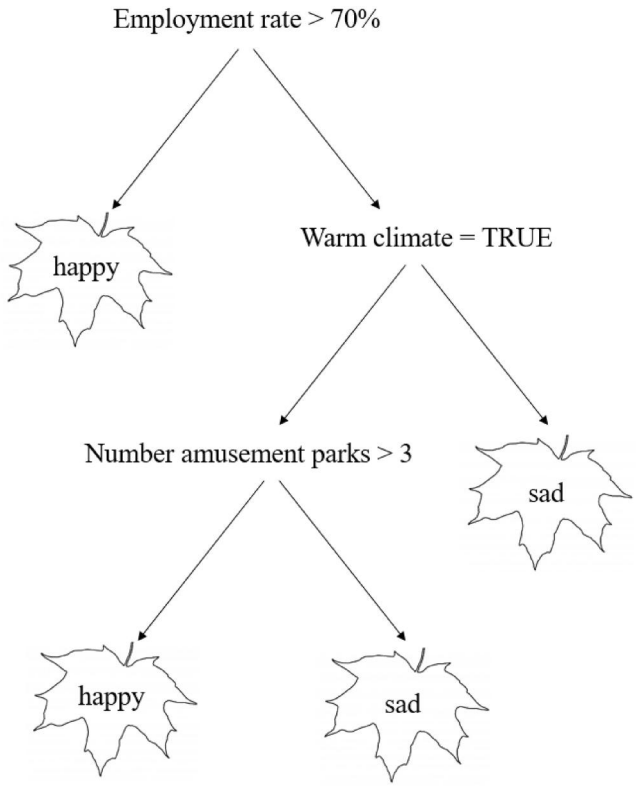


FIGURE 4 A fictitious decision tree model used to predict county-level happiness. Each observation has scores on employment rate, warm climate, and a number of amusement parks. The final "leaves" at the bottom of the tree give the model's prediction for new cases

$$\text{Entropy}_i = -\frac{\#\text{happy}_i}{\#\text{allcases}_i} \times \log\left(\frac{\#\text{happy}_i}{\#\text{allcases}_i}\right) - \frac{\#\text{sad}_i}{\#\text{allcases}_i} \times \log\left(\frac{\#\text{sad}_i}{\#\text{allcases}_i}\right)$$

The entropy of the overall model is a weighted sum of the leaves' entropies. The sequence of questions in the tree is determined by the information gain, which quantifies the decrease in entropy after an additional data split. At each new node, the binary question which provides the highest information gain is selected and appended to the tree. There are alternatives to using the information entropy approach, most notably the Gini index method (e.g., Breiman, 2017), which can have advantages in terms of computational costs.

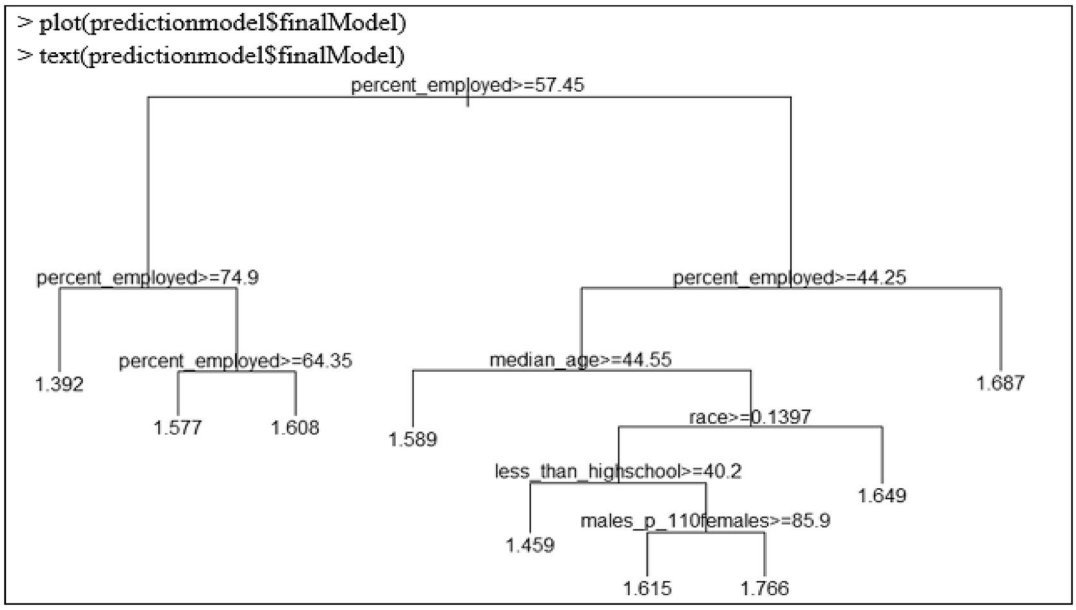
Very large trees (with many splits) are more successful in minimizing entropy and prediction residuals in the training data. However, they run an increased risk of overfitting. Imagine a tree that keeps adding data splits until each leaf only consists of a single training case. Such a tree has perfect accuracy on the training set but generalizes poorly to new samples. Setting a maximum tree size allows researchers to control this trade-off. Hyperparameters that can be used for this purpose are the maximum "number of leaves" or "number of data splits." Here, we give an example of tuning the slightly more sophisticated hyperparameter *cp* (the complexity parameter), which determines how high the minimal increase in R^2 has to be for a new branch to be drawn.

3.2.1 | Implementation of decision trees

In the code example, we predict regional happiness based on common census variables, including regional levels of education, gender ratio, median age, unemployment, ratios of White and Black people, proportions of democratic and republican voters, and population size. To show an alternative method of tuning hyperparameters, we do not explicitly set the values for *cp* in the code example, but let caret pick values at random. The number of values to be generated can be set through the argument "tuneLength." We repeat the steps of splitting off test data, fitting models on training data, and selecting the best model on the development data 20 times, so we can estimate the variability in our accuracy evaluation. Alternatively, researchers can implement a nested cross-validation procedure (see Kuhn & Johnson, n.d.).

```
> #create an empty vector to store the accuracies
> achieved_accuracies <- c()

> #repeat splitting, fitting, and evaluating 20 times
> for(i in 1:20){

> #data splitting
> indices <- caret::createDataPartition(data$happiness, p = 0.75, list=FALSE)
> train_dev_data <- data[indices,]
> testdata <- data[-indices,]

> #set up cross-validation
> cross_validation <- trainControl(method="cv", number=10)

> #train models with 8 different hyperparameters
> predictionmodel <- train(happiness ~ ., data = train_dev_data, method = 'rpart',tuneLength = 8,
trControl = cross_validation)

> #use best model to make predictions on test set
> test_predictions <- predict(predictionmodel, testdata)
> test_result <- R2(test_predictions, testdata$happiness)

> #append the achieved accuracy (20 in total)
> achieved_accuracies <- c(achieved_accuracies, test_result)
}
There were 20 warnings (use warnings() to see them)

> #inspect results across 20 iterations
> summary(achieved_accuracies)
   Min.   1st Qu.    Median    Mean   3rd Qu.    Max.
  0.023   0.088     0.117     0.109   0.126      0.202
```

There are warnings because caret sometimes sets the *cp* parameter so high that no binary split of the data can achieve the desired increase in R^2 (resulting in an empty decision tree). As shown, the best models achieved an average accuracy of $R^2 = .11$. Decision tree models can also be visualized in tree form in the caret package.



Note. The split into counties with high and low employment rates is done first (at the top of the tree) as it is the most informative (helps the most to distinguish relatively happy and relatively sad counties). After this initial split, the employment variable is still the most informative as it is used again for splitting, etc.

As depicted in the graph, most predictor variables are excluded as they do not enable improvements higher than the *cp* hyperparameter dictates.

3.2.2 | Random forest models

While the decision tree algorithm is well-known, its prediction accuracy can virtually always be improved through models that build on its basic structure. The most prominent extension is the random forest model, which consists of many different decision trees (therefore called "model ensemble"). Random forest models make predictions by averaging predictions of its trees (continuous outcome) or by selecting their most common prediction (categorical outcome). Why are the individual trees in the forest different from each other? When building each decision tree, a random (bootstrapped) sample of available observations is selected, and a random subset of available predictor variables is considered for each split. Having many different decision trees based on slightly different sets of observations and predictor variables minimizes the biases of individual trees and reduces overfitting. Thus, it is often not necessary to limit the size of the individual trees in the random forest (Breiman, 2001). Notice that the bootstrap resampling approach inherent to random forests allows users to specify yet another technique for out-of-sample validation often called out-of-bag validation. The according out-of-sample score is computed by assessing the accuracy of decision trees on cases that were not included in the bootstrap sample used to build the tree. In the caret package, users can obtain these scores by specifying the trainControl method as "oob." The disadvantages of this method are that scores cannot be easily compared with nontree-based models, the oob score could be confused with the biased training accuracy (depending on software), and accuracy scores are only computed with a subsample of trees (which did not contain the corresponding test sample).

There are two primary hyperparameters to be set when fitting a random forest model. First, the number of decision trees has to be predetermined. As higher numbers of trees do not lead to overfitting and allow for better predictions, it is common to set this hyperparameter to a high number (e.g., 500; Oshiro, Perez, & Baranauskas, 2012). Increasing the number of trees does not have disadvantages, apart from increased computational costs. Second, the number of predictor variables that get considered at each split must be set (hyperparameter "mtry" below). Including more variables at any stage can lead to either overfitting or better predictions, which

means that this value must be tuned more carefully. It is common to select the square root of the number of available predictors and systematically investigate how higher and lower values affect the performance.

3.2.3 | Implementation of random forests

In the code example, we predict regional happiness using the predictor variables introduced in the decision tree example. Before, we used the outer loop only to quantify prediction accuracy (and the inner loop to select the best hyperparameter). Now, we also keep track of the best hyperparameter chosen in each of the 20 iterations. Additionally, we compare the accuracies of the random forest model and the decision tree model through a paired samples *t*-test (Huang, Lu, & Ling, 2003). Inferential tests to compare model performance are common in machine learning (Salzberg, 1997), but of smaller importance than assessing the practical significance of accuracy differences.

```
> #create empty vectors to store the accuracies and best hyperparameters across iterations
> achieved_accuracies2 = c()
> best_mtry = c()

> #20 iterations
> for(i in 1:20){

> #split data
> indices <- caret::createDataPartition(data$happiness, p = 0.75, list=FALSE)
> train_dev_data <- data[indices,]
> testdata <- data[-indices,]

> #set up cross-validation
> cross_validation = trainControl(method="cv", number=10)

> #set up hyperparameter
> tuning = expand.grid(mtry = c(1,2,3,5,7,9))

> #train models
> predictionmodel = train(happiness ~ ., data = train_dev_data, method = 'rf', tuneGrid = tuning,
trControl = cross_validation)

> #make and evaluate predictions with best model
> test_predictions = predict(predictionmodel, testdata)
> test_result = R2(test_predictions, testdata$happiness)

> #append achieved accuracy
> achieved_accuracies2 = c(achieved_accuracies2, test_result)

> #append best hyperparameter
> best_mtry = c(best_mtry, predictionmodel$bestTune$mtry)}

> #inspect accuracies and hyperparameters across iterations
> summary(achieved_accuracies2)
 Min.   1st Qu.  Median    Mean   3rd Qu.  Max.
 0.112  0.157   0.177   0.179   0.194   0.264
> table(best_mtry)
best_mtry
 2  3  5  7
 2 12  5  1
```

Notice, that when calling "predict" with the random forest model, each tree makes a prediction, but the output constitutes the average of all these predictions. Had the outcome variable been a dichotomous measure of happiness, the prediction would have been the majority prediction from all decision trees. The most accurate model (based on the inner cross-validation) most often considers three randomly selected predictor variables at each split. This model improves the accuracy achieved previously with the decision tree model by a substantial margin.

```
> #compare accuracies with decision tree results
> t.test(achieved_accuracies, achieved_accuracies2, paired = TRUE, alternative = "two.sided")
t = -6.1189, df = 19, p-value = 6.982e-06
95 percent confidence interval:
 -0.094 -0.046
mean of the differences
        -0.070
```

On the 20 iterations, the random forest provides an average increase of 7% in explained variance, which is an increase of 57% relative to the decision tree models. Inferential tests comparing the sets of achieved accuracies confirm that the superiority of the random forest models is not only practical but also statistically, significant ($t(19) = 6.119$, $p < .001$).

Yet, the final random forest model is not easily interpretable, as it consists of 500 distinct trees. A traditional OLS regression model based on the same predictors achieved an average accuracy of $R^2 = .11$ on the test data. Example implementations of decision trees and forests are provided by a range of authors in various subdisciplines of psychology (Joel et al., 2017; Piper, Loh, Smith, Japuntich, & Baker, 2011; Plonsky et al., 2017). When trying to further improve the accuracy of random forest models for high-dimensional, collinear data, researchers often apply predictor selection methods, which filter the useful predictor variables and discard the others from model building. Various algorithms for such predictor selection (including links to R code) have been compared by Degenhardt, Seifert, and Szymczak (2017). One way to assess the relative importance of predictors in random forest models is to rank the predictors in terms of their average information gain (as described in the section on decision trees).

We discussed models based on linear combinations and dichotomizations of predictor variables. In the supplementary material, we provide guidance for alternative modeling options based on point distances, Bayes' rule, and predictor space transformations. These additional models help to illustrate some of the diverse approaches used in machine learning research.

4 | DISCUSSION

Traditional psychological research aims to establish causal effects of predictor variables on outcome variables, whereas machine learning projects aim to achieve maximal (unbiased) accuracy when predicting outcome variables. Still, the intentions of researchers in both disciplines often converge. Psychologists are also interested in the question of how well A predicts B, and therefore the number of papers using machine learning is growing.

Our code examples demonstrate that traditional regression models are frequently less accurate in prediction than alternative models. By applying the methods and tools of machine learning, psychology can better serve society as a predictive science (Yarkoni & Westfall, 2017).

4.1 | Limitations of machine learning

Critics of machine learning models have argued that they are black-boxy, atheoretical, or too complex to allow human interpretation (Krause, Perer, & Ng, 2016). While this is true for some of the most sophisticated models, it

applies less so too many commonly used models (Hindman, 2015). Machine learning includes a range of clustering methods ("unsupervised learning"), which allow for the detection of theoretically meaningful patterns in psychological data (Jordan & Mitchell, 2015). Experimental psychologists can use machine learning techniques to explore differences between experimental conditions (Koul, Becchio, & Cavallo, 2018), and personality researchers can use them to assess the validity of psychological constructs (Bleidorn & Hopwood, 2019). Thus, while we concentrated on prediction accuracy, machine learning also has much to offer to theory-driven research.

Similarly, it is sometimes argued that machine learning methods need too much data and their application is unrealistic for many areas of psychology. Again, the required sample size depends on the context and the model that researchers utilize. For instance, when using image data, models tend to be very complex, and one might need a million images for building a new model from scratch. Conversely, when only two or three predictor variables are available, say from an online survey, a linear regression model will usually require less than 1000 observations to reach maximum performance.

Of course, the field of machine learning also has methodological struggles: Some issues, such as insufficient data sharing, reporting, and replication are akin to challenges in psychological research (Hutson, 2018). Other issues are more characteristic of machine learning, including social biases in predictive models (Veale & Binns, 2017) or unequal distribution of computational resources (Amolo, 2018).

4.2 | Best practices

With a new set of methods and tools, there come new mistakes that can be made. Therefore, we discuss five issues that authors and reviewers should pay attention to in psychological machine learning research. First, prediction accuracy is accompanied by a degree of uncertainty. While a single split into training and test set gives exact numbers of achieved accuracy, the next split usually shows a different accuracy. This instability is especially large when the sample size is relatively small and the model includes many coefficients. Repeating the splitting process usually leads to a more reliable estimation. However, the estimate remains uncertain and quantifications of uncertainty (e.g., through confidence intervals) are needed.

Second, the data left-out for final model testing should not inform the model (i.e., there should be no "information leak"). Common dangers are to preselect predictor variables based on their correlation with the outcome *before* splitting the data into training and test sets. Such practices lead to an artificially inflated measure of accuracy. To avoid such biases, all model characteristics should be selected before the model is evaluated on the final testing data. This should be done by selecting the best predictors, hyperparameters, and models on dedicated development sets. An optimal procedure is to pre-register the final model in a public repository, collect new data, and report the accuracy that the pre-registered model achieved on the new data.

Third, the authors should not selectively report accuracy metrics. For example, models sometimes have seemingly small mean absolute errors, whereas the proportion of explained variance is negligible. This occurs when the outcome variable is tightly clustered around its mean value and making predictions with small errors is therefore easy. Reporting complementary metrics and baseline accuracies prevents misinterpretations.

Fourth, when comparing the predictive value of two competing sets of predictor variables, multiple models should be considered. It is likely that the predictor sets are not equally compatible with all available models. Thus, fitting two regression models and finding that one predictor set leads to higher accuracy does not imply that this set is always more useful; it merely demonstrates that the set is more useful when using linear regression. Results may differ when using other, potentially more accurate models. Relatedly, practical considerations might steer model selection. A deep neural network, which requires immense computational resources might not be the optimal choice if less costly models perform almost as well.

Fifth, common questionable research practices can be exacerbated in machine learning projects. For example, machine learning models are often preceded by multiple preprocessing steps (e.g., standardizing predictors for

ridge regression). Transparency is needed to evaluate and replicate prediction accuracies. One of the most powerful techniques to guard oneself against many of the listed mistakes is to follow the necessities of open science. For machine learning projects, this includes pre-registering and publishing predictions models (through data and code) and making them openly available for review and replication.

4.3 | Additional topics in machine learning

Topics that were either only briefly or not at all discussed include additional cross-validation techniques (Kohavi, 1995), boosting (Dietterich, 2000), reinforcement learning (Sutton & Barto, 2018), deep learning (LeCun, Bengio, & Hinton, 2015), and advanced data preprocessing/acquisition steps (e.g., for unbalanced data; Chawla, Japkowicz, & Kotcz, 2004). These topics warrant a review of their own, but psychologists interested in applying machine learning themselves certainly benefit from familiarizing themselves with these concepts. Introductions to machine learning in marketing (Brei, 2020; Kübler et al., 2017) and economics (Athey & Imbens, 2019) are also of value for psychologists as well as general introductions, as provided by Berk (2006; 2008).

5 | CONCLUSION

We provide researchers in psychology with concrete guidance on implementing and reviewing machine learning research. We highlighted machine learning's focus on generating accurate prediction models. Further, we introduced the main metrics to quantify prediction accuracy, as well as different strategies to evaluate these accuracies on new data. Relatedly, we described the practice of tuning machine learning models through hyperparameters, and selecting the best hyperparameter settings on dedicated development sets, before quantifying the final model's achieved accuracy. Further, we introduced some of the most common machine learning models, alongside annotated implementations in R code. Finally, we discussed some dangers and questionable practices for implementing machine learning models in psychological research. Together, we hope that the current review and tutorial sections will facilitate research aiming to predict psychological phenomena.

ORCID

Hannes Rosenbusch https://orcid.org/0000-0002-4983-3615

ENDNOTES

[1] Supervised machine learning refers to cases where the goal is to predict some known outcome variable. In contrast, unsupervised machine learning approaches refer to problems related to clustering (i.e., identifying the underlying structure in a dataset).

[2] The accuracy is substantially higher when excluding counties with unreliable scores (i.e., where predictor and outcome only have a handful of measurements).

[3] A further variable appearing in the code is called "alpha," which is not typically part of ridge regression. Here, we set alpha to zero, which simply tells the more general "glmnet" model that we want to compute a ridge regression model. An alternative approach would be to directly set the method argument to "ridge" instead of "glmnet," which would allow us to leave out the alpha specification. However, this method appears to take more computational resources based on our test runs, which might reflect differences in the methods' back-end implementation.

[4] Subscript *i* refers to the *i*th leaf. All variables in the formula are counts (e.g., $happy_i$ = number of happy training cases on leaf *i*).

REFERENCES

Akosa, J. S. (2017). Predictive accuracy: A misleading performance measure for highly imbalanced data. In: *Proceedings of the SAS Global Forum*.

Alharthi, R., Guthier, B., & El Saddik, A. (2018). Recognizing human needs during critical events using machine learning powered psychology-based framework. *IEEE Access*, *6*, 58737–58753.

Amolo, G. O. (2018). The growth of high-performance computing in Africa. *Computing in Science & Engineering*, *20*, 21–24.

Askin, R. G. (1982). Multicollinearity in regression: Review and examples. *Journal of Forecasting*, *1*, 281–292.

Athey, S., & Imbens, G. W. (2019). Machine learning methods that economists should know about. *Annual Review of Economics*, *11*, 685–725.

Bachrach, Y., Graepel, T., Kohli, P., Kosinski, M., & Stillwell, D. (2014). Your digital image: Factors behind demographic and psychometric predictions from social network profiles. In: *13th International Conference on Autonomous Agents and Multiagent Systems AAMAS 2014*.

Bellos, S., Skapinakis, P., Rai, D., Zitko, P., Araya, R., Lewis, G., & Mavreas, V. (2013). Cross-cultural patterns of the association between varying levels of alcohol consumption and the common mental disorders of depression and anxiety: Secondary analysis of the WHO Collaborative Study on Psychological Problems in General Health Care. *Drug and Alcohol Dependence*, *133*, 825–831.

Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, *13*, 281–305.

Berk, R. A. (2006). An introduction to ensemble methods for data analysis. *Sociological Methods & Research*, *34*, 263–295.

Berk, R. A. (2008). *Statistical learning from a regression perspective*. New York: Springer, 14.

Bleidorn, W., & Hopwood, C. J. (2019). Using machine learning to advance personality assessment and theory. *Personality and Social Psychology Review*, *23*(2), 190–203.

Brandt, M. J. (2017). Predicting ideological prejudice. *Psychological Science*, *28*, 713–722.

Brei, V. A. (2020). Machine learning in marketing. *Foundations and Trends in Marketing*, *14*, 173–236.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*, 5–32.

Breiman, L. (2017). *Classification and regression trees*. Milton, Oxfordshire: Routledge.

BRFSS. (2005-2010). *SMART data and documentation*. Retrieved from https://www.cdc.gov/brfss/smart/Smart_data.htm

Chawla, N. V., Japkowicz, N., & Kotcz, A. (2004). Special issue on learning from imbalanced data sets. *ACM Sigkdd Explorations Newsletter*, *6*, 1–6.

Chekroud, A. M., Zotti, R. J., Shehzad, Z., Gueorguieva, R., Johnson, M. K., Trivedi, M. H., … Corlett, P. R. (2016). Cross-trial prediction of treatment outcome in depression: A machine learning approach. *The Lancet Psychiatry*, *3*, 243–250.

Claesen, M., & De Moor, B. (2015). *Hyperparameter search in machine learning*. Retrieved from https://arxiv.org/pdf/1502.02127v2.pdf

Dana, J., & Dawes, R. M. (2004). The superiority of simple alternatives to regression for social science predictions. *Journal of Educational and Behavioral Statistics*, *29*, 317–331.

Degenhardt, F., Seifert, S., & Szymczak, S. (2017). Evaluation of variable selection methods for random forests and omics data sets. *Briefings in Bioinformatics*, *20*(2), 492–503.

Dietterich, T. G. (2000). An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, *40*, 139–157.

Eichstaedt, J. C., Schwartz, H. A., Kern, M. L., Park, G., Labarthe, D. R., Merchant, R. M., & Seligman, M. E. P. (2015). Psychological language on Twitter predicts county-level heart disease mortality. *Psychological Science*, *26*, 159–169.

Figueroa, R. L., Zeng-Treitler, Q., Kandula, S., & Ngo, L. H. (2012). Predicting sample size required for classification performance. *BMC Medical Informatics and Decision Making*, *12*, 8–18.

Ghandeharioun, A., Fedor, S., Sangermano, L., Ionescu, D., Alpert, J., Dale, C., & Picard, R. (2017). Objective assessment of depressive symptoms with machine learning and wearable sensors data. In: *Seventh International Conference on Affective Computing and Intelligent Interaction (ACII)*.

Golbeck, J., Robles, C., & Turner, K. (2011). Predicting personality with social media. In: *CHI'11 extended abstracts on human factors in computing systems* (pp. 253–262) ACM.

Goldfeld, K. (2018). *simstudy: Simulation of Study Data. R package version 0.1.10*. Retrieved from https://CRAN.R-project.org/package=simstudy

Hassan, L. M., Shiu, E., & Shaw, D. (2016). Who says there is an intention–behaviour gap? Assessing the empirical evidence of an intention–behaviour gap in ethical consumption. *Journal of Business Ethics*, *136*, 219–236.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 770–778).

Hindman, M. (2015). Building better models: Prediction, replication, and machine learning in the social sciences. *The Annals of the American Academy of Political and Social Science*, *659*, 48–62.

Huang, J., Lu, J., & Ling, C. X. (2003, November). Comparing naive Bayes, decision trees, and SVM with AUC and accuracy. *Null* (pp. 553). IEEE.

Hutson, M. (2018). Artificial intelligence faces reproducibility crisis. *Science*, *359*(6377), 725–726.

Jie, M. A., Collins, G. S., Steyerberg, E. W., Verbakel, J. Y., & van Calster, B. (2019). A systematic review shows no performance benefit of machine learning over logistic regression for clinical prediction models. *Journal of Clinical Epidemiology*, *110*, 12–22.

Joel, S., Eastwick, P. W., & Finkel, E. J. (2017). Is romantic desire predictable? Machine learning applied to initial romantic attraction. *Psychological Science*, *28*, 1478–1489.

Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, *349*(6245), 255–260.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In: *Proceedings of the 14th International Joint Conference on Artificial Intelligence* (pp. 1137–1143).

Kosinski, M., Bachrach, Y., Kohli, P., Stillwell, D., & Graepel, T. (2014). Manifestations of user personality in website choice and behaviour on online social networks. *Machine Learning*, *95*, 357–380.

Koul, A., Becchio, C., & Cavallo, A. (2018). PredPsych: A toolbox for predictive machine learning-based approach in experimental psychology research. *Behavior Research Methods*, *50*, 1–16.

Krause, J., Perer, A., & Ng, K. (2016). Interacting with predictions: Visual inspection of black-box machine learning models. In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (pp. 5686–5697).

Kübler, R. V., Colicev, A., & Pauwels, K. H. (2020). Social media's impact on the consumer mindset: When to use which sentiment extraction tool?. *Journal of Interactive Marketing*, *50*, 136–155.

Kübler, R., Wieringa, J. E., & Pauwels, K. H. (2017). Machine learning and big data. In P. S. Leeflang, J. E. Wieringa, T. H. Bijmolt, & K. H. Pauwels (Eds.), *Advanced methods for modeling markets* (pp. 631–670). Berlin: Springer.

Kuhn, M. (2015). *A short introduction to the caret package* R Foundation for Statistical Computing.

Kuhn, M., & Johnson, K. (2013). *Applied predictive modeling*. New York, NY: Springer, 26.

Kuhn, M., & Johnson, K. *Nested resampling with rsample*. Retrieved from http://appliedpredictivemodeling.com/blog/2017/9/2/njdc83d01pzysvvlgik02t5qnaljnd

Kuhn, M., & Wickham, H. (2020). *Tidymodels: A collection of packages for modeling and machine learning using tidyverse principles* Retrieved from https://www.tidymodels.org

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444.

Lever, J., Krzywinski, M., & Altman, N. (2016). Points of significance: Model selection and overfitting. *Nature Methods*, *13*, 703–704.

Macmillan, N. A. (2002). Signal detection theory. In H. Pashler & J. Wixted (Eds.), *Stevens' handbook of experimental psychology: Methodology in experimental psychology* (pp. 43–90). Hoboken, NJ: John Wiley & Sons Inc.

Murphy, K. P. (2012). *Machine learning: A probabilistic perspective*. Cambridge, MA: MIT Press.

Ng, A. (2018). *Size of the dev and test sets* Retrieved from https://www.coursera.org/lecture/machine-learning-projects/size-of-the-dev-and-test-sets-HOby4

Oshiro, T. M., Perez, P. S., & Baranauskas, J. A. (2012). How many trees in a random forest? In: *International Workshop on Machine Learning and Data Mining in Pattern Recognition* (pp. 154–168). Berlin, Heidelberg: Springer.

Park, G., Schwartz, H., Eichstaedt, J. C., Kern, M. L., Kosinski, M., Stillwell, D. J., & Seligman, M. E. P. (2015). Automatic personality assessment through social media language. *Journal of Personality and Social Psychology*, *108*, 934–952.

Piper, M. E., Loh, W. Y., Smith, S. S., Japuntich, S. J., & Baker, T. B. (2011). Using decision tree analysis to identify risk factors for relapse to smoking. *Substance Use & Misuse*, *46*, 492–510.

Plonsky, O., Erev, I., Hazan, T., & Tennenholtz, M. (2017). Psychological forest: Predicting human behavior. In: *The Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)* (pp. 656–662).

R Core Team. (2018). *R: A language and environment for statistical computing*. Vienna, Austria: R Foundation for Statistical Computing Retrieved from https://www.R-project.org/

Raudys, S. J., & Jain, A. K. (1991). Small sample size effects in statistical pattern recognition: Recommendations for practitioners. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *13*, 252–264.

Ribeiro, J. D., Pease, J. L., Gutierrez, P. M., Silva, C., Bernert, R. A., Rudd, M. D., & Joiner, T. E., Jr. (2012). Sleep problems outperform depression and hopelessness as cross-sectional and longitudinal predictors of suicidal ideation and behavior in young adults in the military. *Journal of Affective Disorders*, *136*, 743–750.

Rimfeld, K., Kovas, Y., Dale, P. S., & Plomin, R. (2016). True grit and genetics: Predicting academic achievement from personality. *Journal of Personality and Social Psychology*, *111*, 780–789.

Salzberg, S. L. (1997). On comparing classifiers: Pitfalls to avoid and a recommended approach. *Data Mining and Knowledge Discovery*, *1*, 317–328.

Scherer, S., Stratou, G., Gratch, J., & Morency, L. P. (2013). Investigating voice quality as a speaker-independent indicator of depression and PTSD. In: *Proceedings of the Annual Conference of the International Speech Communication Association* INTERSPEECH.

Schwartz, H. A., Eichstaedt, J. C., Kern, M. L., Dziurzynski, L., Ramones, S. M., Agrawal, M., & Ungar, L. H. (2013). Personality, gender, and age in the language of social media: The open-vocabulary approach. *PLoS One*, *8*(9), e73791.

Sumner, C., Byers, A., Boochever, R., & Park, G. J. (2012). Predicting dark triad personality traits from twitter usage and a linguistic analysis of tweets. In: *Proceedings - 2012 11th International Conference on Machine Learning and Applications ICMLA 2012, 2*.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. Boston, MA: MIT Press.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B*, *58*(1), 267–288.

Veale, M., & Binns, R. (2017). Fairer machine learning in the real world: Mitigating discrimination without collecting sensitive data. *Big Data & Society*, *4*, 1–17.

Walsh, C. G., Ribeiro, J. D., & Franklin, J. C. (2017). Predicting risk of suicide attempts over time through machine learning. *Clinical Psychological Science*, *5*, 457–469.

Wang, N., Kosinski, M., Stillwell, D. J., & Rust, J. (2014). Can well-being be measured using Facebook status updates? Validation of Facebook's gross national happiness index. *Social Indicators Research*, *115*, 483–491.

Wang, Y., & Kosinski, M. (2018). Deep neural networks are more accurate than humans at detecting sexual orientation from facial images. *Journal of Personality and Social Psychology*, *114*, 246–257.

Yarkoni, T., & Westfall, J. (2017). Choosing prediction over explanation in psychology: Lessons from machine learning. *Perspectives on Psychological Science*, *12*, 1100–1122.

Youyou, W., Kosinski, M., & Stillwell, D. (2015). Computer-based personality judgments are more accurate than those made by humans. *Proceedings of the National Academy of Sciences*. 112 (pp. 1036–1040).

Zhang, T., & Oles, F. J. (2001). Text categorization based on regularized linear classification methods. *Information Retrieval*, *4*, 5–31.

AUTHOR BIOGRAPHY

Hannes Rosenbusch is a PhD candidate at Tilburg University. He works on integrating data science techniques, like machine learning and advanced data processing, into psychological research.

How to cite this article: Rosenbusch H, Soldner F, Evans AM, Zeelenberg M. Supervised machine learning methods in psychology: A practical introduction with annotated R code. *Soc Personal Psychol Compass*. 2021;15:e12579. https://doi.org/10.1111/spc3.12579